МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Череповецкий государственный университет»

**Лабораторная работа № 2**

**«Сортировка массивов»**

**Выполнил:**

студент гр. 1ИВТпб-01-31оп

Климов А.Г.  
**Проверил:** преподаватель

Пышницкий К.М.  
Отметка о зачете:

Череповец

2017 год

**Задание:** Составить программу, проводящую сравнительную характеристику методов сортировки массивов.

Программа должна выполнять следующие действия:

1. Производить сортировку массива соответствующими методами.

2. Иллюстрировать работу каждого метода на небольших массивах (размером до 10 элементов).

3. Производить сортировку каждым из методов случайного массива, уже отсортированного массива, массива, отсортированного в обратном порядке. Размер массива при этом должен выбираться пользователем. После проведения сортировки, вывести данные о скорости работы методов.

**Вариант:** 5.Метод прямого включения, шейкерная сортировка, метод Шелла.

**Текст программы:**

**MyForm.h**

#pragma once

#include <windows.h>

#include <string>

#include "time.h"

#include <iostream>

//=======Основные переменные=======//

std::string Number = ""; //Cтрока для вывода результата на экран

std::string Result =""; //Cтрока для вывода результата на экран

int start\_time\_sort = 0; // Начальное время таймера

int end\_time\_sort = 0; // Конечное время таймера

int n=0;

//=======Объявление функций=======//

void Insertion\_sort(int a[], int n); // Метод прямого включения

void Shaker\_sort(int a[], int n); // Шейкерная сортировка

void Shella\_sort(int a[], int n); // Метод Шелла

void print(int a[], int n); // Вывод массива

void time(); // Время работы сортировки

//========Описание функций========//

void Insertion\_sort(int a[], int n)

{

start\_time\_sort = clock();

for (int i = 2; i < n; i++)

{

int k = a[i];

a[0] = k;

int j = i;

while (k < a[j - 1]) { a[j] = a[j - 1]; j--; }

a[j] = k;

}

end\_time\_sort = clock();

}

void Shaker\_sort(int a[], int n)

{

start\_time\_sort = clock();

int l = 1, r = n - 1, k = n - 1;

do

{

for (int j = r; j >= l; j--)

if (a[j] < a[j - 1])

{

std::swap(a[j], a[j - 1]);

k = j;

}

l = k + 1;

for (int j = l; j <= r; j++)

if (a[j] < a[j - 1]) { std::swap(a[j], a[j - 1]); k = j; }

r = k - 1;

} while (l <= r);

end\_time\_sort = clock();

}

void Shella\_sort(int a[], int n)

{

start\_time\_sort = clock();

int incr = n / 2;

while (incr > 0)

{

for (int i = incr; i < n; i++) {

int j = i - incr;

while (j >= 0)

if (a[j] > a[j + incr]) { std::swap(a[j], a[j + incr]); j -= incr; }

else j = -1;

}

incr /= 2;

}

end\_time\_sort = clock();

}

void print(int a[], int n)

{

for (int i = 0; i < n; i++)

{

Result += std::to\_string(a[i]);

Result += " ";

}

}

void print\_insertion(int a[], int n)

{

for (int i = 1; i < n; i++)

{

Result += std::to\_string(a[i]);

Result += " ";

}

}

void time()

{

Result += "\nВремя работы: ";

Result += std::to\_string(end\_time\_sort - start\_time\_sort);

Result += " мс";

}

namespace Project1 {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

/// <summary>

/// Summary for MyForm

/// </summary>

public ref class MyForm : public System::Windows::Forms::Form

{

public:

MyForm(void)

{

InitializeComponent();

//

//TODO: Add the constructor code here

//

}

protected:

/// <summary>

/// Clean up any resources being used.

/// </summary>

~MyForm()

{

if (components)

{

delete components;

}

}

private: System::Windows::Forms::Button^ button1;

private: System::Windows::Forms::RichTextBox^ richTextBox1;

private: System::Windows::Forms::Button^ button2;

private: System::Windows::Forms::TextBox^ textBox1;

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Button^ button3;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::Label^ label5;

private: System::Windows::Forms::Button^ button4;

private: System::Windows::Forms::Label^ label3;

protected:

private:

/// <summary>

/// Required designer variable.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

void InitializeComponent(void)

{

this->button1 = (gcnew System::Windows::Forms::Button());

this->richTextBox1 = (gcnew System::Windows::Forms::RichTextBox());

this->button2 = (gcnew System::Windows::Forms::Button());

this->textBox1 = (gcnew System::Windows::Forms::TextBox());

this->label1 = (gcnew System::Windows::Forms::Label());

this->button3 = (gcnew System::Windows::Forms::Button());

this->label2 = (gcnew System::Windows::Forms::Label());

this->label4 = (gcnew System::Windows::Forms::Label());

this->label5 = (gcnew System::Windows::Forms::Label());

this->button4 = (gcnew System::Windows::Forms::Button());

this->label3 = (gcnew System::Windows::Forms::Label());

this->SuspendLayout();

//

// button1

//

this->button1->Location = System::Drawing::Point(517, 296);

this->button1->Name = L"button1";

this->button1->Size = System::Drawing::Size(75, 23);

this->button1->TabIndex = 1;

this->button1->Text = L"Ввод";

this->button1->UseVisualStyleBackColor = true;

this->button1->Click += gcnew System::EventHandler(this, &MyForm::button1\_Click);

//

// richTextBox1

//

this->richTextBox1->Location = System::Drawing::Point(23, 13);

this->richTextBox1->Name = L"richTextBox1";

this->richTextBox1->ReadOnly = true;

this->richTextBox1->Size = System::Drawing::Size(569, 235);

this->richTextBox1->TabIndex = 2;

this->richTextBox1->Text = L"";

this->richTextBox1->TextChanged += gcnew System::EventHandler(this, &MyForm::richTextBox1\_TextChanged);

//

// button2

//

this->button2->Location = System::Drawing::Point(517, 267);

this->button2->Name = L"button2";

this->button2->Size = System::Drawing::Size(75, 23);

this->button2->TabIndex = 3;

this->button2->Text = L"Очистить";

this->button2->UseVisualStyleBackColor = true;

this->button2->Click += gcnew System::EventHandler(this, &MyForm::button2\_Click);

//

// textBox1

//

this->textBox1->Location = System::Drawing::Point(405, 269);

this->textBox1->Name = L"textBox1";

this->textBox1->Size = System::Drawing::Size(76, 20);

this->textBox1->TabIndex = 4;

this->textBox1->TextChanged += gcnew System::EventHandler(this, &MyForm::textBox1\_TextChanged);

this->textBox1->KeyPress += gcnew System::Windows::Forms::KeyPressEventHandler(this, &MyForm::textBox1\_KeyPress);

//

// label1

//

this->label1->AutoSize = true;

this->label1->Location = System::Drawing::Point(303, 272);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(96, 13);

this->label1->TabIndex = 5;

this->label1->Text = L"Размер массива:";

this->label1->Click += gcnew System::EventHandler(this, &MyForm::label1\_Click);

//

// button3

//

this->button3->Location = System::Drawing::Point(517, 325);

this->button3->Name = L"button3";

this->button3->Size = System::Drawing::Size(75, 23);

this->button3->TabIndex = 6;

this->button3->Text = L"Ввод";

this->button3->UseVisualStyleBackColor = true;

this->button3->Click += gcnew System::EventHandler(this, &MyForm::button3\_Click);

//

// label2

//

this->label2->AutoSize = true;

this->label2->Location = System::Drawing::Point(302, 301);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(209, 13);

this->label2->TabIndex = 7;

this->label2->Text = L"Демонстрация работы каждого метода";

this->label2->Click += gcnew System::EventHandler(this, &MyForm::label2\_Click);

//

// label4

//

this->label4->AutoSize = true;

this->label4->Location = System::Drawing::Point(303, 324);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(210, 13);

this->label4->TabIndex = 9;

this->label4->Text = L"Сортировка отсортированного массива";

this->label4->Click += gcnew System::EventHandler(this, &MyForm::label4\_Click);

//

// label5

//

this->label5->AutoSize = true;

this->label5->Location = System::Drawing::Point(303, 348);

this->label5->Name = L"label5";

this->label5->Size = System::Drawing::Size(213, 26);

this->label5->TabIndex = 10;

this->label5->Text = L"Сортировка массива, отсортированного\r\nв обратном порядке";

this->label5->Click += gcnew System::EventHandler(this, &MyForm::label5\_Click);

//

// button4

//

this->button4->Location = System::Drawing::Point(517, 354);

this->button4->Name = L"button4";

this->button4->Size = System::Drawing::Size(75, 23);

this->button4->TabIndex = 11;

this->button4->Text = L"Ввод";

this->button4->UseVisualStyleBackColor = true;

this->button4->Click += gcnew System::EventHandler(this, &MyForm::button4\_Click);

//

// label3

//

this->label3->AutoSize = true;

this->label3->Location = System::Drawing::Point(20, 272);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(251, 104);

this->label3->TabIndex = 12;

this->label3->Text = L"Программа, проводящая сравнительную\r\nхарактеристику методов сортировки массивов.\r"

L"\n\r\n5 Вариант:\r\n метод прямого включения;\r\n шейкерная сортировка;\r\n метод Шелла;\r"

L"\n\r\n";

//

// MyForm

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(604, 395);

this->Controls->Add(this->label3);

this->Controls->Add(this->button4);

this->Controls->Add(this->label5);

this->Controls->Add(this->label4);

this->Controls->Add(this->label2);

this->Controls->Add(this->button3);

this->Controls->Add(this->label1);

this->Controls->Add(this->textBox1);

this->Controls->Add(this->button2);

this->Controls->Add(this->richTextBox1);

this->Controls->Add(this->button1);

this->FormBorderStyle = System::Windows::Forms::FormBorderStyle::Fixed3D;

this->MaximizeBox = false;

this->MinimizeBox = false;

this->Name = L"MyForm";

this->Text = L"Климов А.Г.";

this->Load += gcnew System::EventHandler(this, &MyForm::MyForm\_Load);

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void MyForm\_Load(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void button1\_Click(System::Object^ sender, System::EventArgs^ e) {

if (textBox1->Text != "") // Проверка на наличие размера массива в textBox1

{

n = Convert::ToInt32(textBox1->Text);

if (n <= 10000) {

// Объявлениие массивов

int z = n + 1;

int \*a = new int[z];

int \*b = new int[n];

int \*c = new int[n];

a[0] = -1;

// Заполнение массива для вставки с барьером

for (int i = 1; i < z; i++)

{

a[i] = rand() % 100;

}

// Заполнение массивов случайными числами

for (int i = 0; i < n; i++)

{

b[i] = a[i+1];

c[i] = a[i+1];

}

// Вывод сформированного массива

Result += "Сфорормирован массив из ";

Result += std::to\_string(n);

Result += " элементов:\n";

print(b, n);

// Сортировки

Insertion\_sort(a, z);

Result += "\n\nМетод прямого включения:\n";

print\_insertion(a, z);

time();

Shaker\_sort(b, n);

Result += "\n\nШейкерный метод:\n";

print(b, n);

time();

Shella\_sort(c, n);

Result += "\n\nМетод Шелла:\n";

print(c, n);

time();

Result += "\n\n";

// Удаление массивов

delete a;

delete b;

delete c;

String^ Result\_converted = gcnew String(Result.c\_str());

richTextBox1->Text = Result\_converted;

}

else MessageBox::Show("Размер массива больше 10000!", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Exclamation);

}

else MessageBox::Show("Размер массива не задан!", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Exclamation);

}

private: System::Void button2\_Click(System::Object^ sender, System::EventArgs^ e) {

Result = "";

richTextBox1->Clear();

}

private: System::Void label1\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void textBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void textBox1\_KeyPress(System::Object^ sender, System::Windows::Forms::KeyPressEventArgs^ e)

{

if (((e->KeyChar >= '0') && (e->KeyChar <= '9'))||e->KeyChar==8) // Если вводятся цифры от 0 до 9 или нажимается backspace то "далее"

return;

e->Handled = true; // Остальные символы запрещены для ввода

}

private: System::Void button3\_Click(System::Object^ sender, System::EventArgs^ e) {

if (textBox1->Text != "") // Проверка на наличие размера массива в textBox1

{

n = Convert::ToInt32(textBox1->Text);

if (n <= 10000) {

// Объявлениие массивов

int z = n + 1;

int \*a1 = new int[z];

int \*b1 = new int[n];

int \*c1 = new int[n];

a1[0] = -1;

// Заполнение массива для вставки с барьером

for (int i = 1; i < z; i++)

{

a1[i] = i-1;

}

// Заполнение массивов их порядковыми номерами

for (int i = 0; i < n; i++)

{

b1[i] = i;

c1[i] = i;

}

// Сортировки

Insertion\_sort(a1, z);

Result += "Метод прямого включения:";

//print(a1, n);

time();

Shaker\_sort(b1, n);

Result += "\n\nШейкерный метод:";

//print(b1, n);

time();

Shella\_sort(c1, n);

Result += "\n\nМетод Шелла:";

//print(c1, n);

time();

Result += "\n\n";

// Удаление массивов

delete a1;

delete b1;

delete c1;

String^ Result\_converted = gcnew String(Result.c\_str());

richTextBox1->Text = Result\_converted;

}

else MessageBox::Show("Размер массива больше 10000!", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Exclamation);

}

else MessageBox::Show("Размер массива не задан!", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Exclamation);

}

private: System::Void label2\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void label4\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void label5\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void button4\_Click(System::Object^ sender, System::EventArgs^ e) {

if (textBox1->Text != "") // Проверка на наличие размера массива в textBox1

{

n = Convert::ToInt32(textBox1->Text);

if (n <= 10000) {

// Объявлениие массивов

int z = n + 1;

int \*a11 = new int[z];

int \*b11 = new int[n];

int \*c11 = new int[n];

a11[0] = -1;

// Заполнение массива для вставки с барьером

for (int i = 1; i < z; i++)

{

a11[i] = z-i;

}

// Заполнение массивов их порядковыми номерами в обратном порядке

for (int i = 0; i < n; i++)

{

b11[i] = n-i;

c11[i] = n-i;

}

// Сортировки

Insertion\_sort(a11, z);

Result += "Метод прямого включения:";

//print(a1, n);

time();

Shaker\_sort(b11, n);

Result += "\n\nШейкерный метод:";

//print(b1, n);

time();

Shella\_sort(c11, n);

Result += "\n\nМетод Шелла:";

//print(c1, n);

time();

Result += "\n\n";

// Удаление массивов

delete a11;

delete b11;

delete c11;

String^ Result\_converted = gcnew String(Result.c\_str());

richTextBox1->Text = Result\_converted;

}

else MessageBox::Show("Размер массива больше 10000!", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Exclamation);

}

else MessageBox::Show("Размер массива не задан!", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Exclamation);

}

private: System::Void richTextBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e) {

}

};

}

**Результаты тестирования:**

Данные о скорости работы методов

Таблица 1

|  |  |  |  |
| --- | --- | --- | --- |
|  | Рандомный массив (100 эл.) | Отсортированный массив (100 эл.) | Отсортированный обратный массив (100 эл.) |
| Метод прямого включения | 0 мс | 0 мс | 0 мс |
| Шейкерная сортировка | 0 мс | 0 мс | 0 мс |
| Метод Шелла | 0 мс | 0 мс | 0 мс |

Таблица 2

|  |  |  |  |
| --- | --- | --- | --- |
|  | Рандомный массив (1000 эл.) | Отсортированный массив (1000 эл.) | Отсортированный обратный массив (1000 эл.) |
| Метод прямого включения | 1 мс | 0 мс | 1 мс |
| Шейкерная сортировка | 5 мс | 0 мс | 7 мс |
| Метод Шелла | 1 мс | 0 мс | 0 мс |

Таблица 3

|  |  |  |  |
| --- | --- | --- | --- |
|  | Рандомный массив (10000 эл.) | Отсортированный массив (10000 эл.) | Отсортированный обратный массив (10000 эл.) |
| Метод прямого включения | 59 мс | 1 мс | 120 мс |
| Шейкерная сортировка | 495 мс (472 мс) | 0 мс | 838 мс (775 мс) |
| Метод Шелла | 2 мс | 0 мс | 1 мс |